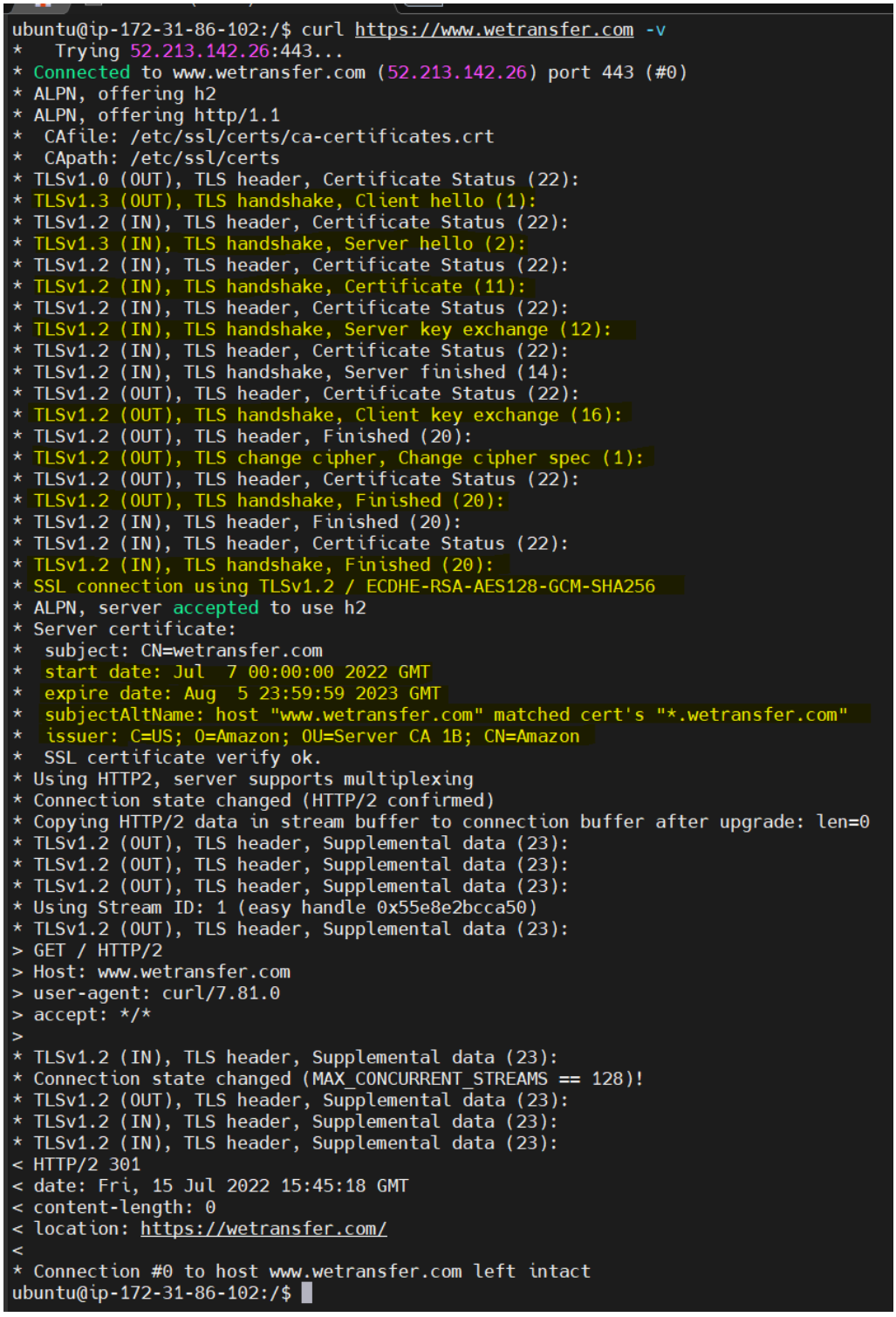
Answer 1

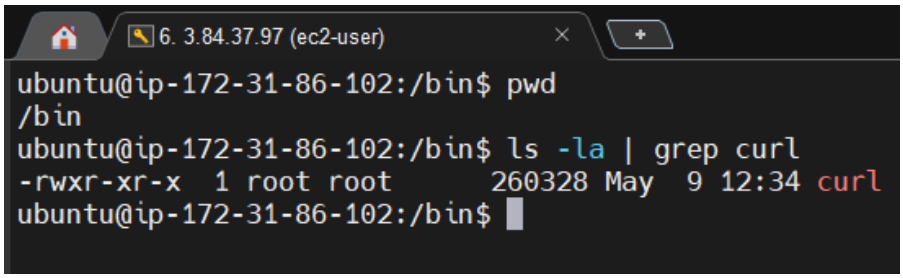
Console Output:



1. We start off with the command itself. It is used to make HTTP & HTTPS requests to an endpoint. Curl more commonly used in unix systems supports HTTP, HTTPS, FTP, SCP and many more protocols.

2. The command get picked up and interpreted by the shell, it changes if the keywords being used is a program contained in the shell itself.

3. The built-in program of curl which is present in /bin/curl (taken from $PATH/curl) is executed and passed on to the kernel to initiate communication with the NIC. (Network Interface Card)



4. The Kernel initiates the communication with NIC through its firmware/drivers and accordingly asks it to perform the tasks.

5. The NIC initiates a UDP call on port 53 to the first DNS server via the router, which usually is your Internet Service Provider or a 3rd Party DNS provider.

6. The host IP Address allocation and the IP address of the DNS resolver is usually configured by the router via DHCP calls when the system connects to the router.

7. Usually the DNS entries for commonly used websites will be found at the first DNS checkpoint (DNS Resolver) itself. Incase the entry is not found then the query is forwarded to the root DNS server, which is in turn forwarded to the .com TLD server which is again forwarded to the domain nameserver which finally returns the IP Address of wetransfer.com. *(In our case, the DNS lookup answer wasn't cached, so our request went all the way to the domain nameserver of wetransfer.com and dns records were retrieved from there, in form of an authoritative response)*

   serial is a revision numbering system for DNS entries, whenever there are any changes, the serial number changes, and secondary nameservers are alerted
   refresh specifies the interval after while secondary DNS server will poll the primary one if there is any change in the dns record
   retry specifies the time length secondary server will wait if primary server is non-responsive.
   expire specifies the time that secondary DNS server will keep cached DNS records valid for, if the primary server doesn't respond after that then the secondary server will stop responding for such queries

```
> set q=soa
> https://www.wetransfer.com
Server:         127.0.0.53
Address:        127.0.0.53#53

Non-authoritative answer:
*** Can't find https://www.wetransfer.com: No answer

Authoritative answers can be found from:
wetransfer.com
        origin = ns-1495.awsdns-58.org
        mail addr = awsdns-hostmaster.amazon.com
        serial = 1
        refresh = 7200
        retry = 900
        expire = 1209600
        minimum = 86400
>
```

```
ubuntu@ip-172-31-86-102:/bin$ nslookup
> https://www.wetransfer.com
Server:         127.0.0.53
Address:        127.0.0.53#53

Non-authoritative answer:
Name:   https://www.wetransfer.com
Address: 52.213.142.26
Name:   https://www.wetransfer.com
Address: 52.214.128.28
Name:   https://www.wetransfer.com
Address: 34.250.122.178
>
```

(Note the highlighted IP, 52.214.128.28)

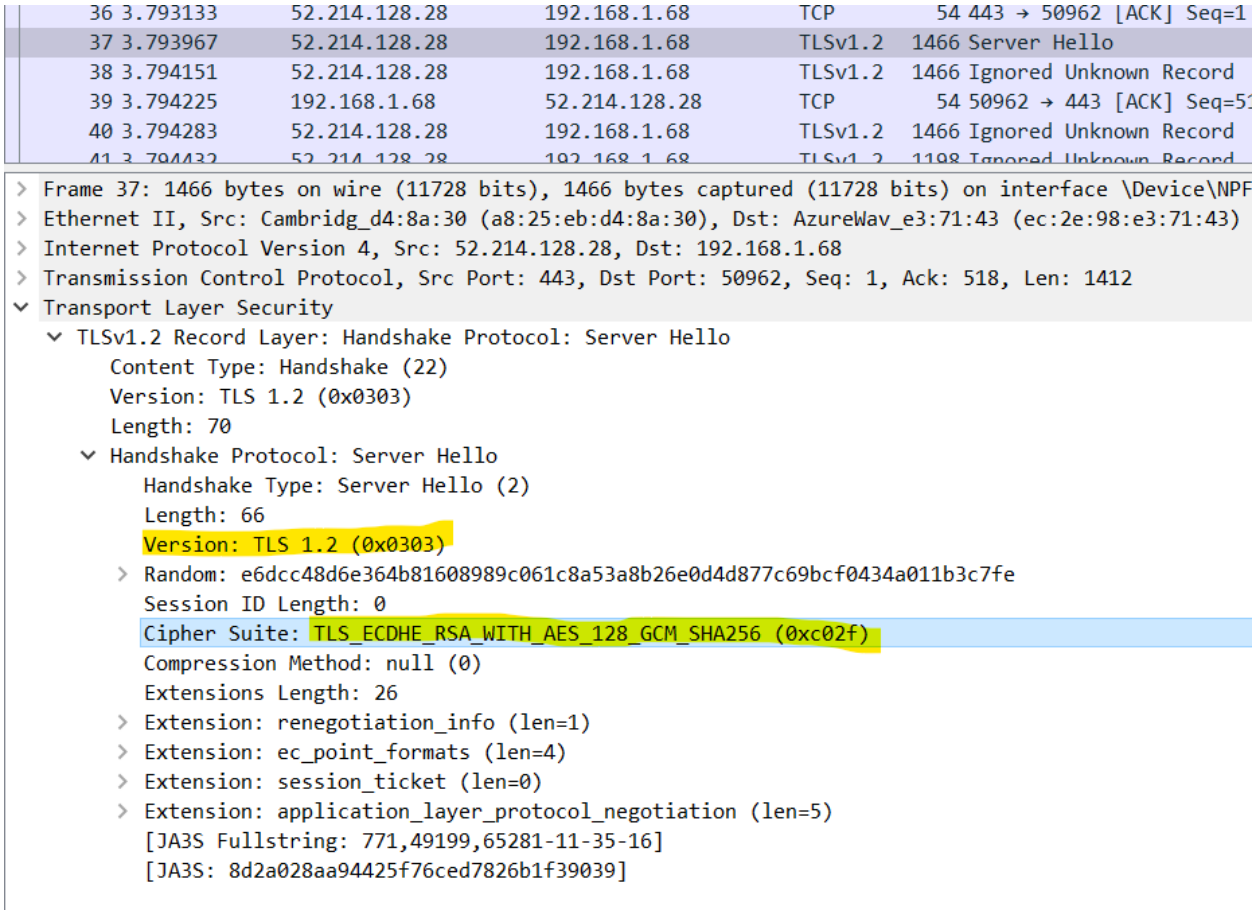
8. The communication starts between the client (curl call from the terminal) and the server. The protocol used will be TCP over port 443 (HTTPS) and it will be a GET request.
9. After establishing the TCP connection and agreeing on the cipher to exchange data, there is a negotiation between the client and the server which is the SSL/TLS handshake.
10. The client sends a hello message packet to the server which contains all the supported ciphers and SSL version. The server then replies with a hello packet of itself, which includes data like the agreed upon cipher to exchange data and SSL version.

Client Hello Packet showing all the list of supported ciphers

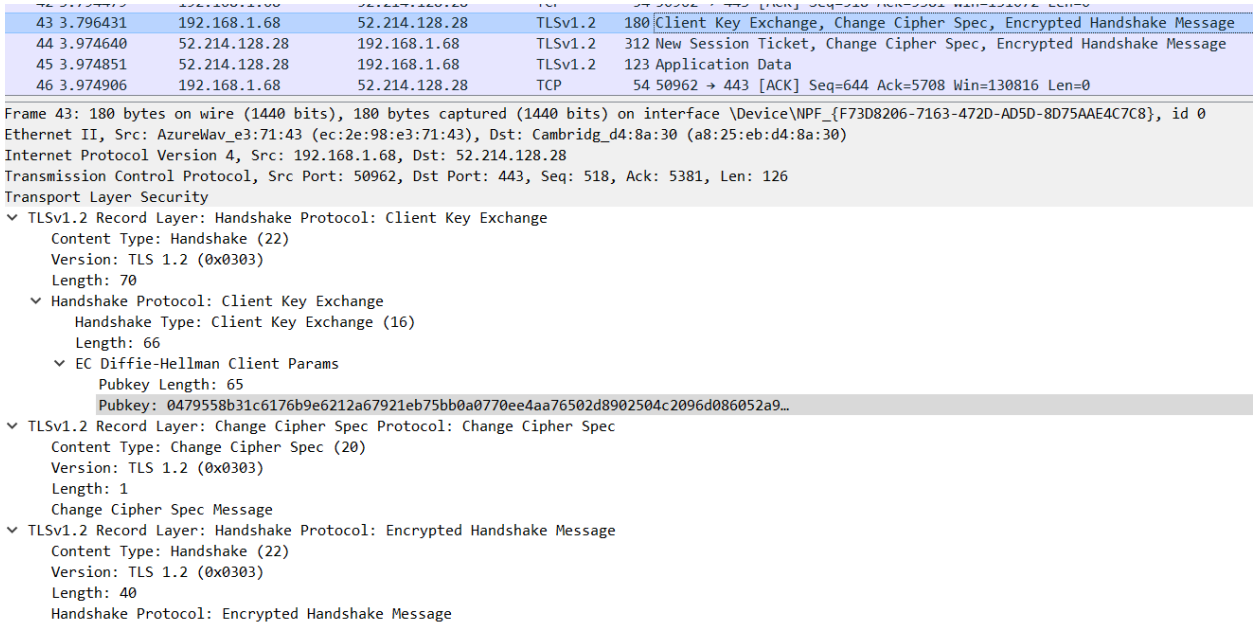
```
35 3.613935    192.168.1.68     52.214.128.28    TLSv1.2    571 Client Hello
36 3.793133    52.214.128.28    192.168.1.68     TCP         54 443 → 50962 [ACK] Seq=1 Ack=518 Win=28160 Len=0
37 3.793967    52.214.128.28    192.168.1.68     TLSv1.2   1466 Server Hello
38 3.794151    52.214.128.28    192.168.1.68     TLSv1.2   1466 Ignored Unknown Record
39 3.794225    192.168.1.68     52.214.128.28    TCP         54 50962 → 443 [ACK] Seq=518 Ack=2825 Win=131072 Len=0
40 3.794283    52.214.128.28    192.168.1.68     TLSv1.2   1466 Ignored Unknown Record
```

```
nsport Layer Security
TLSv1.2 Record Layer: Handshake Protocol: Client Hello
  Content Type: Handshake (22)
  Version: TLS 1.0 (0x0301)
  Length: 512
v Handshake Protocol: Client Hello
    Handshake Type: Client Hello (1)
    Length: 508
    Version: TLS 1.2 (0x0303)
  > Random: 3970a96672eaa843c2e774f362653e5a2fa8889813a29a49a3de605d223e4162
    Session ID Length: 32
    Session ID: a7967a6e9b7cbe44ac4db27814e129c68d494d8611f817d38d385305e06e2142
    Cipher Suites Length: 32
  v Cipher Suites (16 suites)
      Cipher Suite: Reserved (GREASE) (0x5a5a)
      Cipher Suite: TLS_AES_128_GCM_SHA256 (0x1301)
      Cipher Suite: TLS_AES_256_GCM_SHA384 (0x1302)
      Cipher Suite: TLS_CHACHA20_POLY1305_SHA256 (0x1303)
      Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 (0xc02b)
      Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f)
      Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 (0xc02c)
      Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (0xc030)
      Cipher Suite: TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256 (0xcca9)
      Cipher Suite: TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256 (0xcca8)
      Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (0xc013)
      Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014)
      Cipher Suite: TLS_RSA_WITH_AES_128_GCM_SHA256 (0x009c)
      Cipher Suite: TLS_RSA_WITH_AES_256_GCM_SHA384 (0x009d)
      Cipher Suite: TLS_RSA_WITH_AES_128_CBC_SHA (0x002f)
      Cipher Suite: TLS_RSA_WITH_AES_256_CBC_SHA (0x0035)
    Compression Methods Length: 1
  > Compression Methods (1 method)
```

Server Hello Packet highlighting the cipher and the TLS version used

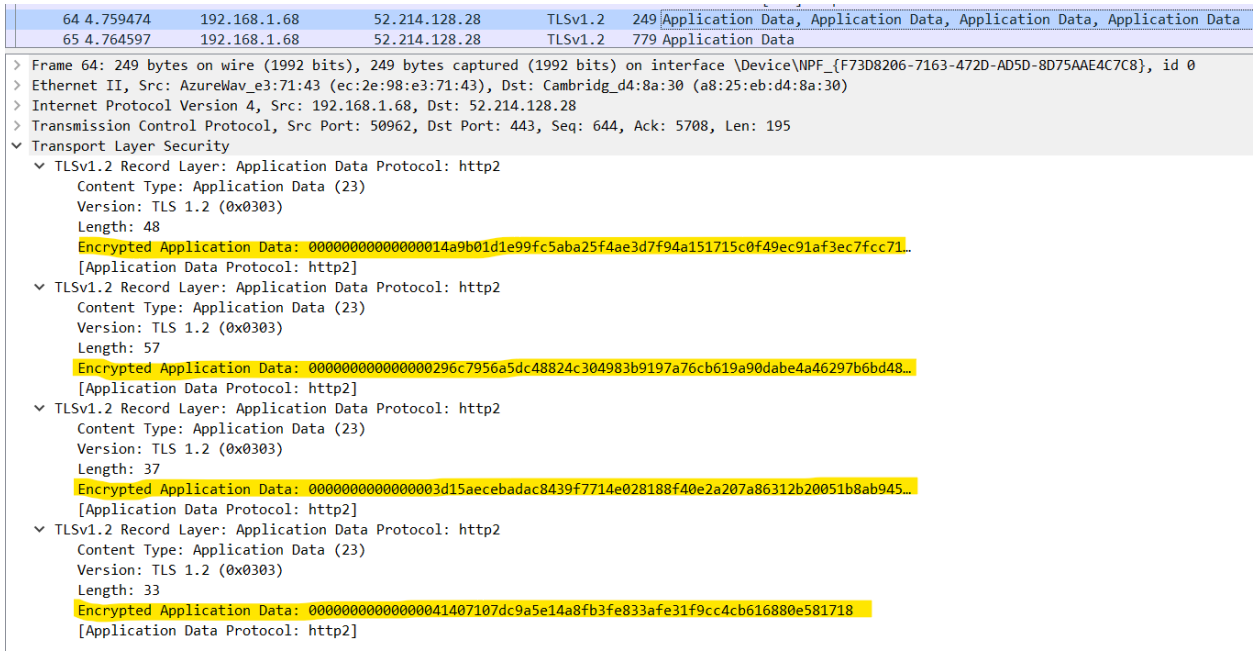


11. Client ensures that it is talking to the right server, by checking the SSL Certificate that is returned by the server which contains the domain name, expiry date and the public key.



12. The encryption of the data is then done by the agreed upon cipher from step #9 alone with the public key for the server which is decrypted at the server end via the server's private key.

<u>Highlighted encrypted data which can be decrypted with server's private key.</u>



13. Once verified an encrypted connection is established between the client and the server, the request is then processed by the server and a response is provided.
14. The response if everything worked correctly, will include a 200 HTTP status code, the content of the website (HTML of the website), headers and other informational data like content size, HTTP Status Description.

```
ubuntu@ip-172-31-86-102:~$ curl https://wetransfer.com -I
HTTP/2 200
date: Fri, 15 Jul 2022 16:09:35 GMT
content-type: text/html; charset=utf-8
cache-control: no-cache, no-store
etag: W/"2bcea18dc5d70a1c1b664a6ec934471c"
expires: Fri, 01 Jan 1990 00:00:00 GMT
pragma: no-cache
referrer-policy: strict-origin-when-cross-origin
set-cookie: _wt_snowplowid.0497=f70c420d-9612-4d63-9c2d-21438d57e
vary: Accept-Encoding
vary: Accept-Encoding, Origin
x-content-type-options: nosniff
x-download-options: noopen
x-frame-options: SAMEORIGIN
x-opaque: c293eaa25aa324648cc6c7fd47396eefe9971b36-k5x66-75096
x-permitted-cross-domain-policies: none
x-request-id: 392183f36efc510e005fcab4c7e8fcee
x-runtime: 0.024516
x-xss-protection: 1; mode=block
strict-transport-security: max-age=15552000; includeSubDomains;

ubuntu@ip-172-31-86-102:~$
```

Answer 2

Please find the required code files and references in this repo.